验证码训练识别流程：

一 准备好训练集与测试集
二 修改 train.py 部分配置进行模型训练
三 修改 test.py 部分配置，查看训练效果

第一部分-准备图片：

1 首先准备好训练和测试集验证码图片，保存在本地目录

训练集

测试集

其中训练集需要至少要有 1000 张以上的验证码图片，保证网络能充分学习。训练集的验证码格式如下：

训练集必须标注验证码标签

第二部分-修改 train.py 部分的代码

2 修改下面的代码的初始化部分

```
62    def __init__(self):
63        # 训练集路径
64        self.train_data_path = './success/'
65        # 测试集路径
66        self.test_data_path ='./fail1/'
67
68        # 写到指定的磁盘路径中
69        self.log_dir = '/home/apps/model7/Tb'
70        # 数据集图片大小
71        self.width = 24
72        self.heigth = 64
73        # 最大迭代次数
74        self.max_steps = 1000000
75        # 读取数据集
76        self.train_imgs,train_labels=self.get_imgs_train()
77        self.test_imgs, self.test_labels = self.get_imgs_test()
78        # 训练集大小
79        self.train_size = len(self.train_imgs)
80        # 测试集大小
81        self.test_size = len(self.test_imgs)
82        # 每次获得batch_size大小的当前训练集指针
83        self.train_ptr = 0
84        # 每次获取batch_size大小的当前测试集指针
85        self.test_ptr = 0
86        # 字符字典大小:0-9 a-z A-Z _(验证码如果小于4, 用_补齐) 一共63个字符
87        self.char_set_len = 63
88        # 验证码最长的长度为4
89        self.max_captcha = 4
90        # 输入数据X占位符
91        self.X = tf.placeholder(tf.float32, [None, self.heigth*self.width])
92        # 输入数据Y占位符
93        self.Y = tf.placeholder(tf.float32, [None, self.char_set_len*self.max_captcha])
94        # keepout占位符
95        self.keep_prob = tf.placeholder(tf.float32)
```

这里需要修改初始化部分第 64，66，69，71，72 行代码，其中 64，66，69 为你自己本机的验证码训练集，测试集保存地址，69 行代码是 log 文件保存地址（可以通过启动 Tensorboard 的方法可视化查看训练过程，具体方法是打开命令行输入 tensorboard –logdir=文件地址，会产生一个网址，在浏览器打开即可查看整个训练过程）。71，72 行根据自己验证码图片大小进行相应修改即可。

3 修改下面这个卷积神经网络函数的代码

```
243    def crack_captcha_cnn(self, w_alpha=0.01, b_alpha=0.1):
244        """
245        定义CNN
246        Parameters:
247            w_alpha:权重系数
248            b_alpha:偏置系数
249        Returns:
250            out:CNN输出
251        """
```

其中，主要修改下面这行代码：

```
308        w_d = tf.Variable(w_alpha*tf.random_normal([3*8*64, 1024]))
```

w_d = tf.Variable(w_alpha*tf.random_normal([**3*8*64**, 1024]))，这里红色部分需要根据上

面的三层卷积后图片输入向量的变化规律作相应修改（不同网站的验证码图片大小不同，作相应修改即可），规律如下：

w_d = tf.Variable(w_alpha*tf.random_normal([（**self.width/8）*（self.heigth/8）*64**, 1024]))，可根据验证码图片的高和宽作计算带入即可。

4 修改下面的网络的训练代码

```
324     def train_crack_captcha_cnn(self):
325         """
326         训练函数
327         """
```

主要修改其中

```
367                     # 如果准确率大于95%，则保存模型并退出。
368                 if acc > 0.95:
369                     train_writer.close()
370                     test_writer.close()
371                     saver.save(sess, "/home/apps/model7/model/"+"crack_capcha.model", global_step=i)
372                     break
```

这里 saver.save(sess, "换成你自己本机的模型文件保存地址"+"crack_capcha.model", global_step=i)，这里其实可以在初始化代码里加个模型地址，以后修改起来更方便。

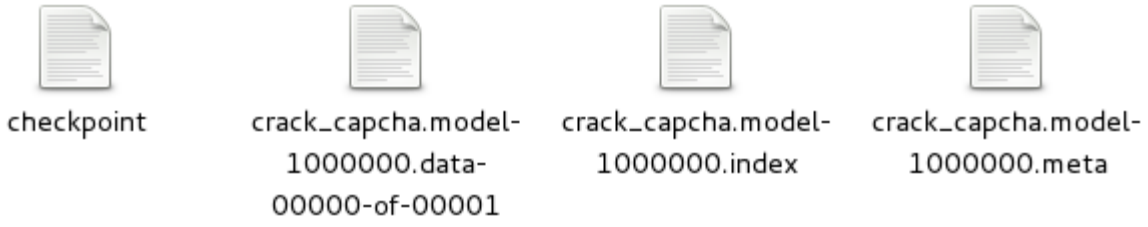
```
373                 # 一直训练
374             else:
375                 batch_x, batch_y = self.get_next_batch(True, 100)
376                 loss_value, _ = sess.run([loss, optimizer], feed_dict={self.X: batch_x, self.Y: batch_y, self.keep
377                 print('迭代第%d次 loss:%f' % (i+1, loss_value))
378                 curve = sess.run(merged, feed_dict={self.X: batch_x_test, self.Y: batch_y_test, self.keep_prob: 1}
379                 train_writer.add_summary(curve, i)
380 
381         train_writer.close()
382         test_writer.close()
383         saver.save(sess, "/home/apps/model7/model/"+"crack_capcha.model", global_step=self.max_steps)
384 
```

同样修改 saver.save()部分，和上面一样把地址换掉。

5 所有修改完成后保存运行 train.py 文件（python train.py）,程序会跑起来，如下图所示：

```
2018-05-07 08:21:11.059757: I tensorflow/core/platfor
迭代第1次 accuracy:0.022500
迭代第2次 loss:0.689443
迭代第3次 loss:0.658346
迭代第4次 loss:0.611593
迭代第5次 loss:0.536826
迭代第6次 loss:0.428539
迭代第7次 loss:0.293564
迭代第8次 loss:0.163689
迭代第9次 loss:0.089334
迭代第10次 loss:0.083317
迭代第11次 accuracy:0.110000
迭代第12次 loss:0.101201
迭代第13次 loss:0.114807
迭代第14次 loss:0.119599
迭代第15次 loss:0.111800
迭代第16次 loss:0.099619
迭代第17次 loss:0.083804
迭代第18次 loss:0.067203
```

大概迭代个 7000 多次，需要跑大概 50 分钟左右，accuracy 会达到 95%左右，程序自动停止，此时打开你自己的建的 model 文件里面会出现你训练的模型，如下图所示：



这就是我们训练出的模型文件。

6 修改 test.py，进行测试，看看效果

首先修改下面这行代码：

```
23    with tf.Session() as sess:
24
25        saver.restore(sess, tf.train.latest_checkpoint("/home/apps/model7/model"))
```

将 saver.restore(sess, tf.train.latest_checkpoint("你自己本机的 model 路径")),这段代码意思是加载你训练的模型。

然后可以修改下面这行代码：

```
39
40 if __name__ == '__main__':
41     dz = train.Ocr()
42     batch_x, batch_y = dz.get_next_batch(False, 20)
```

batch_x, batch_y = dz.get_next_batch(False, 20)，其中 20 可以换成你所期望的数字，这里面 20 代表你想随机在测试集里挑多少个验证码进行测试。

```
2018-05-07 02:33:34.719518: I tensorflow/core/plat
标签： 4009  预测： 4909
标签： 1192  预测： 1192
标签： 6644  预测： 6644
标签： 0649  预测： 0649
标签： 6240  预测： 6240
标签： 4164  预测： 4164
标签： 2991  预测： 2991
标签： 0049  预测： 0049
标签： 6805  预测： 6805
标签： 9455  预测： 9455
标签： 0813  预测： 0813
标签： 3193  预测： 3193
标签： 1716  预测： 1716
标签： 3653  预测： 3653
标签： 6815  预测： 6815
标签： 5337  预测： 5337
标签： 1562  预测： 1562
标签： 8664  预测： 8664
标签： 3751  预测： 3754
标签： 6828  预测： 6828
```

项目全部代码的 gitlab 地址：http://git.epmap.org/jiwei.chen/OCR